Ciel ! Mon Kubernetes mine des

Bitcoins

~$ whoami

Denis GERMAIN

Senior SRE chez deezer

Auteur principal sur blog.zwindler.fr*

@zwindler / @zwindler_rflx

#geek #SF #courseAPied

** Les slides de ce talk sont sur le blog*

deezer en quelques chiffres

- Créé en 2007
- 73M de titres
- 16M d'utilisateurs actifs
- ~35k connections par seconde
- 1 gros monolithe en cours de tronçonnage

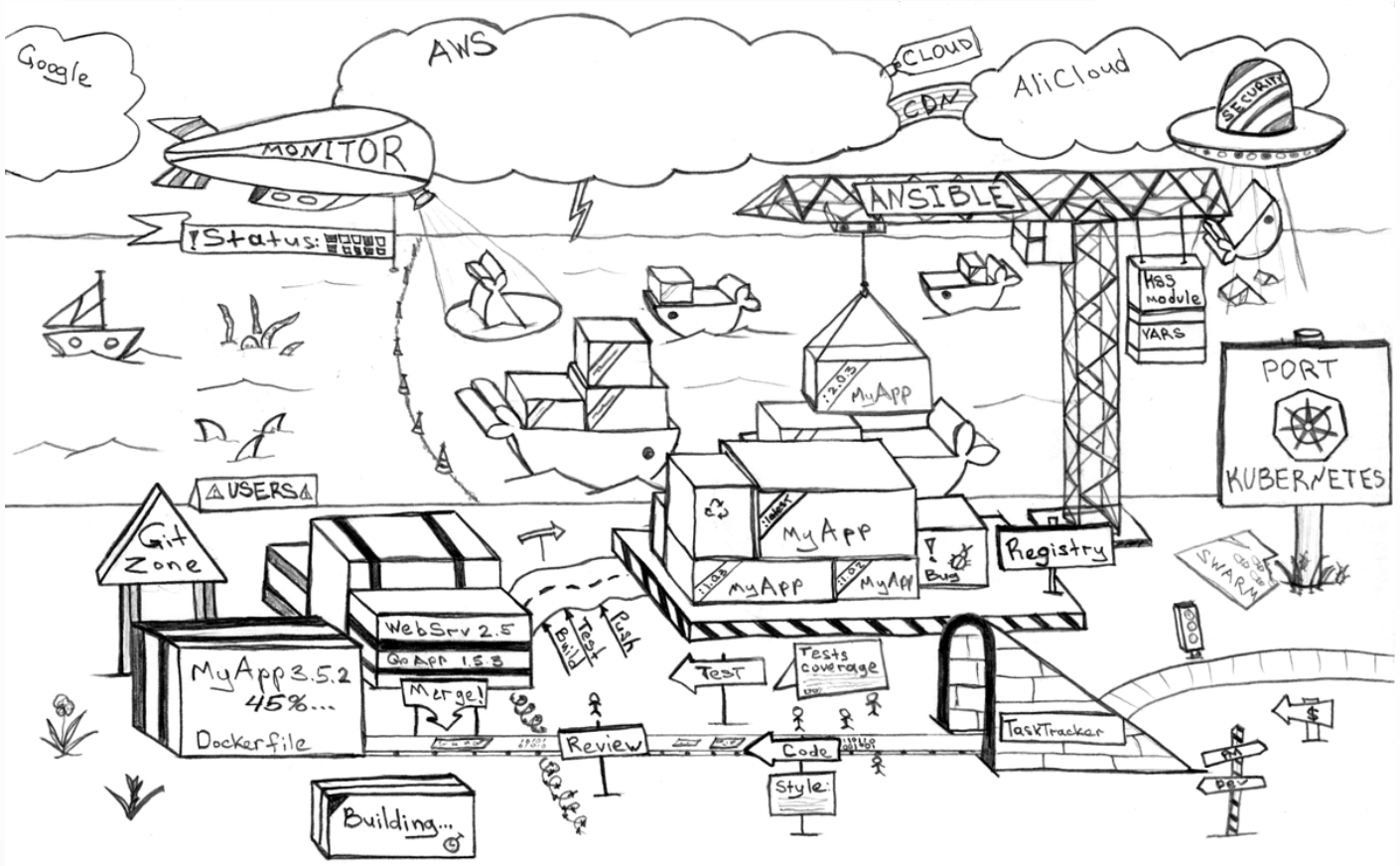
D’ailleurs… je cherche des collègues !

- Un·e Network Engineer SRE pour connecter tout ce binioù 👩‍💻 👨‍💻
- Et des Devs pour ~~casser~~ faire scaler les environnements 🤣
- Et plus encore
- GOTO ➔ deezerjobs (ou demandez moi, je suis gentil)

Ciel ! Mon Kubernetes mine des

~~Bitcoins~~  Monero 

Mais... c'est quoi Kubernetes / Docker déjà ?



Crédits : [Dmitriy Paunin]

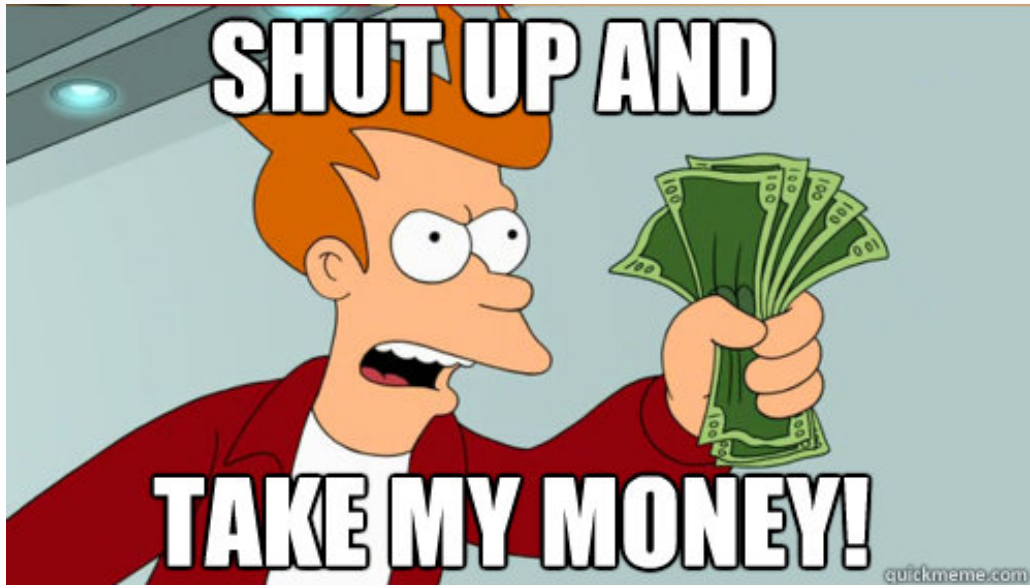
Les containers Docker

Technologie de containerisation d'applications

- sortie en 2013
- utilise des fonctionnalités du kernel Linux
- gestion via une interface "simple"
- fourni un magasin d'images librement accessibles

Docker et ses promesses

- Rend l'infrastructure *facile* pour le développeur
- Economies (par rapport aux VMs)
- Sécurité (isolation des applications)
- Immutabilité (déploiements et mises à jours reproductibles)



Retour à la réalité

Techniquement : on a réinventé les `jail` avec une interface de management "simple" et des (très) gros binaires

Mais on ne sait toujours pas comment gérer :

- **la haute disponibilité ?**
- **la tolérance de panne ?**
- **les droits d'accès ?**

Kubernetes

- Orchestrateur de containers, inspiré par un outil interne de Google
- Donné à la CNCF (spin-off Linux Foundation)
- Open Sourcé en 2015

C'est un outil puissant et complexe

> Kubernetes définit un **certain** nombre d'objets pilotables par API, qui ensemble fournissent des mécanismes pour déployer, maintenir et mettre à l'échelle des applications

- *Pod, Deployment, ReplicaSet, DaemonSet, StatefulSet* pour l'appli
- *Role, RoleBinding, ClusterRoleBinding, ServiceAccount* pour la gestion des droits
- ...

Par extension, c'est un outil verbeux

Lancer nginx dans **Docker**

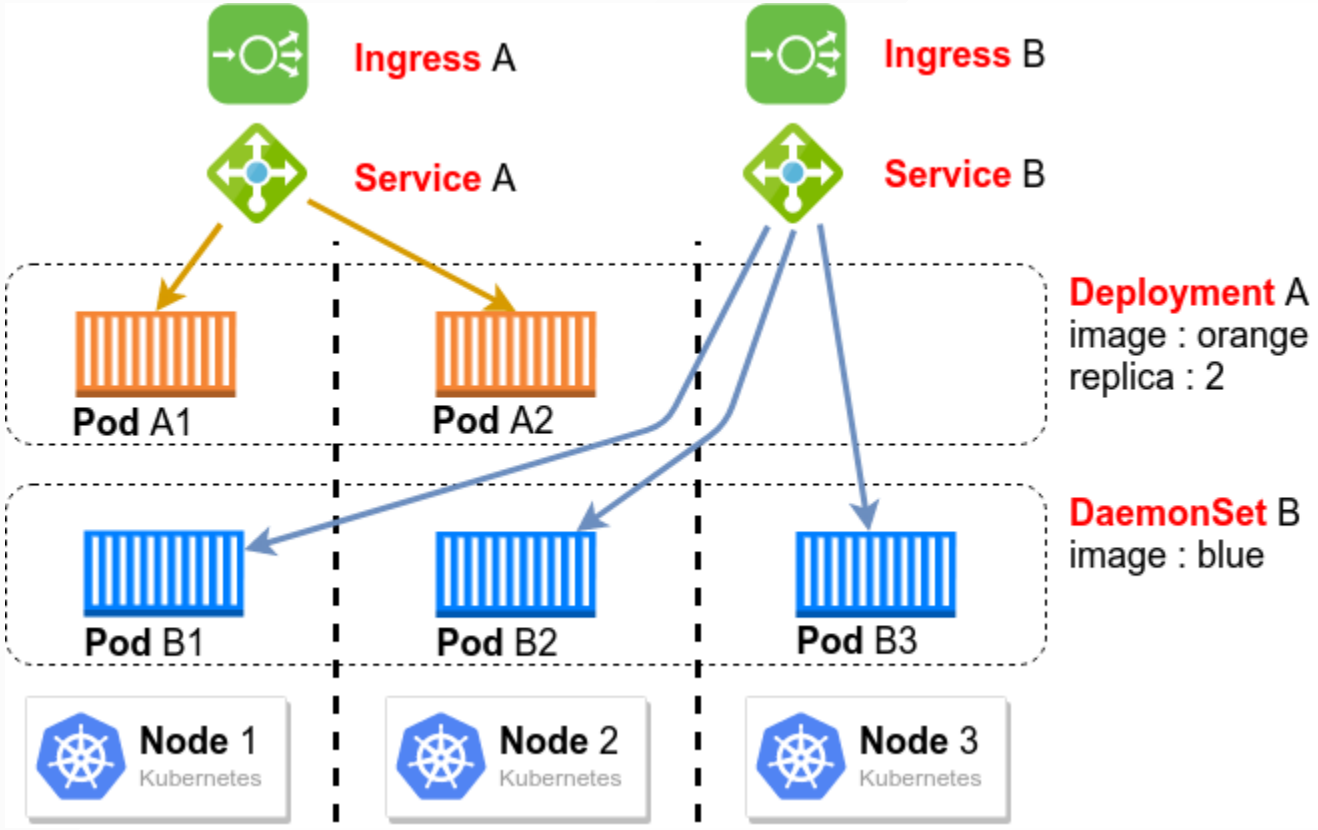
```
~$ docker run --name nginx -d nginx
```

Versus dans **Kubernetes**

```yaml
apiVersion: apps/v1 # for versions before 1.9.0 use apps/v1beta2
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  selector:
    matchLabels:
      app: nginx
  replicas: 2 # tells deployment to run 2 pods matching the template
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.7.9
        ports:
        - containerPort: 80
```

Abstraire l'infrastructure

Décrire l'état souhaité de notre application hautement disponible



What could possibly go wrong ?

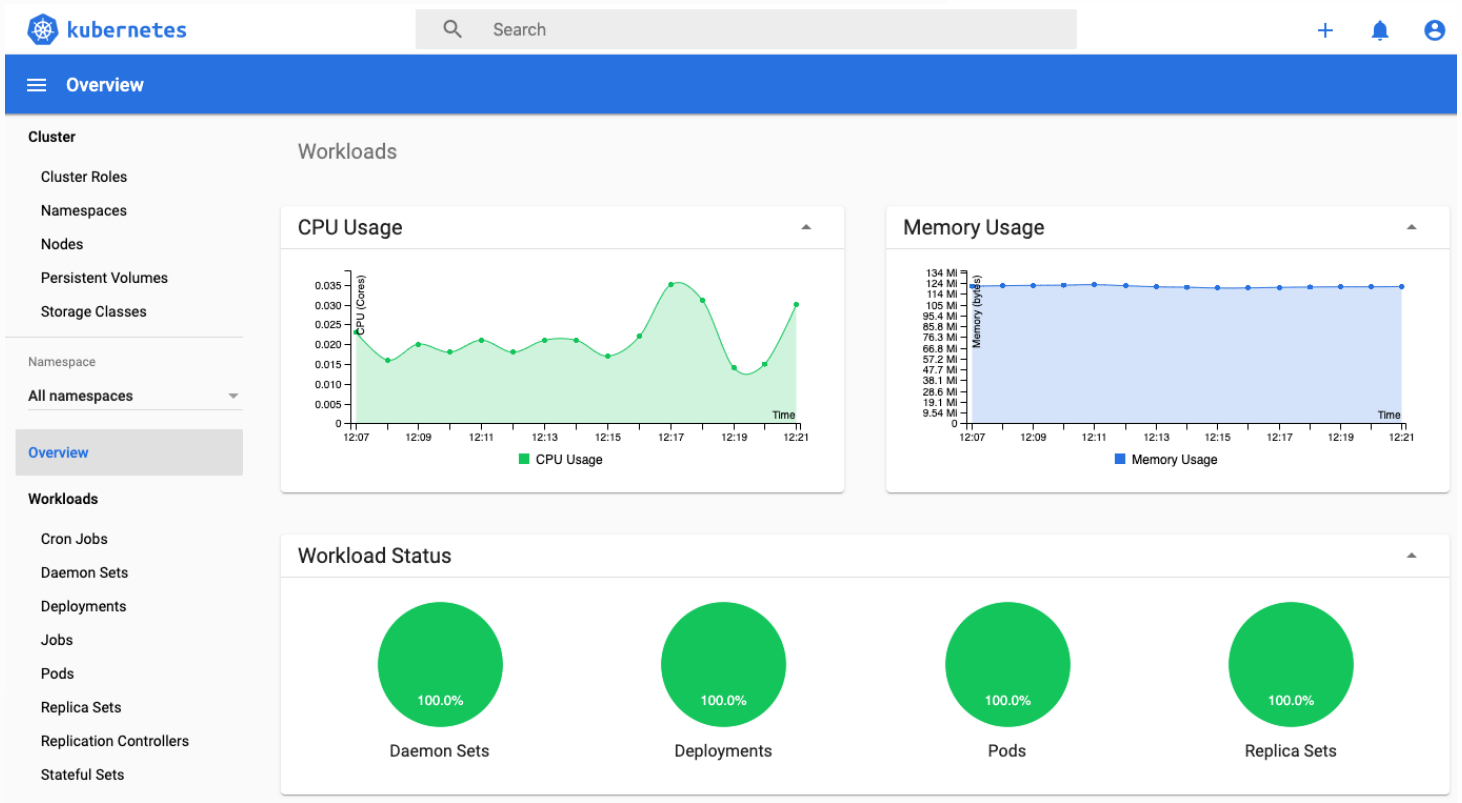
Un outil complexe ? Pas grave, il y a une UI !

L'histoire récente regorge de failles et d'exploits sur des interfaces de management ouvertes sur Internet

- **phpMyAdmin**
- **tomcat-manager**
- **webmin**
- ...

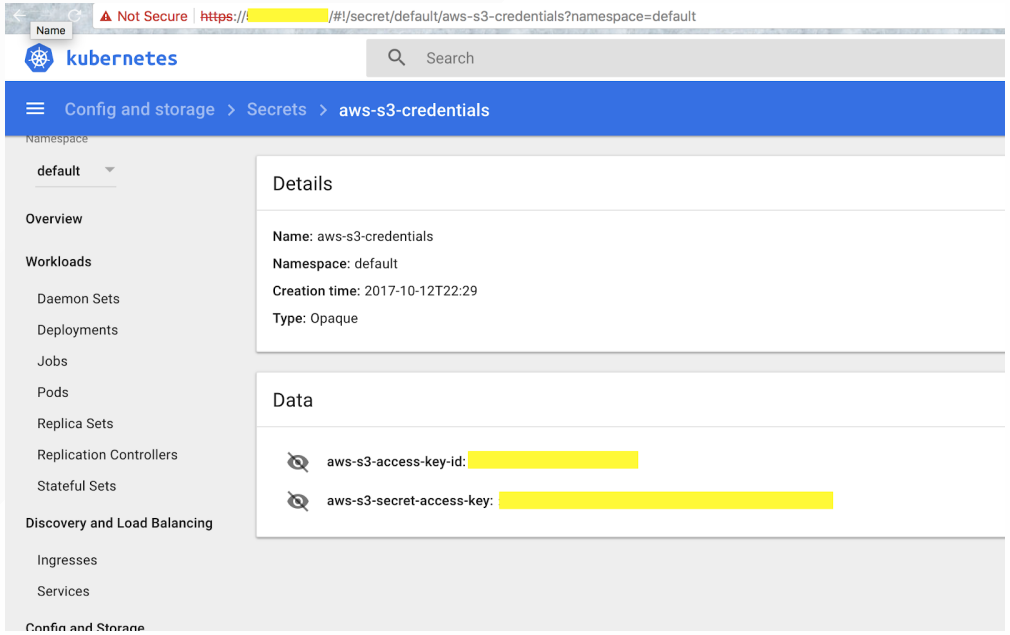
Et pourtant... Tesla T...

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes/dashboard/v2.0.0-beta6/aio/deploy/recommended.yaml
```



“Not password protected”

> The hackers had infiltrated Tesla’s Kubernetes console which was **not password protected** / Source : redlock.io



Moralité : sortez couvert (sur Internet)

Vraiment.

N'exposez pas la console. Si vous ne l'utilisez pas, ne la déployez même pas.

- ~~Vue incomplète de votre cluster et de votre métrologie~~
- Les clouds providers la désactivent par défaut
- Préférez lui
  - pour la gestion : `kubectl` ou des UIs locales
  - pour la métrologie : **Grafana**, **Prometheus**, outils tiers

Autre exemple : Kubeflow

- Dashboard pour planifier des tâches de machine learning
- ML => GPU
- $$$$$$

[zdnet - Microsoft découvre un gang de cryptomining détournant des clusters Kubernetes]

Utilisez une authentification tierce

Pas de gestion des (vrais) utilisateurs. Les applications/démons ont des **ServiceAccounts** authentifiés par :

- Tokens JWT
- Certificats (difficilement révocables 😭)

Ajouter une authentification tierce de type OIDC + RBAC

Contrôle d'accès dans Kubernetes

- Depuis la 1.6 (2017), RBAC (Role-based access control) par défaut
  - Donner des droits fins
    - par type de ressource
    - par type d'accès
  - De les affecter à des groupes d'utilisateurs ou d'applications
- Appliquez le **principe de moindre privilège**

Le RBAC par l'exemple

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  namespace: default
  name: pod-reader
rules:
- apiGroups: [""]
  resources: ["pods"]
  verbs: ["get", "watch", "list"]
```

```yaml
apiVersion: rbac.authorization.k8s.io/v1

kind: RoleBinding
metadata:
  name: read-pods
  namespace: default
subjects:
- kind: User
  name: alice
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: Role
  name: pod-reader
  apiGroup: rbac.authorization.k8s.io
```

Ex. **alice** a le droit de lister les containers dans le namespace **default**, mais pas de les supprimer ni les créer.

En cas de compromission

Si un compte utilisateur/application est compromis, les accès de l'attaquant seront limités à un périmètre donné :

- **Namespace** (subdivision logique du cluster)
- types d'actions précis pour chaque type de ressources

Dans la pratique

Le principe des moindres privilèges est un vrai chantier

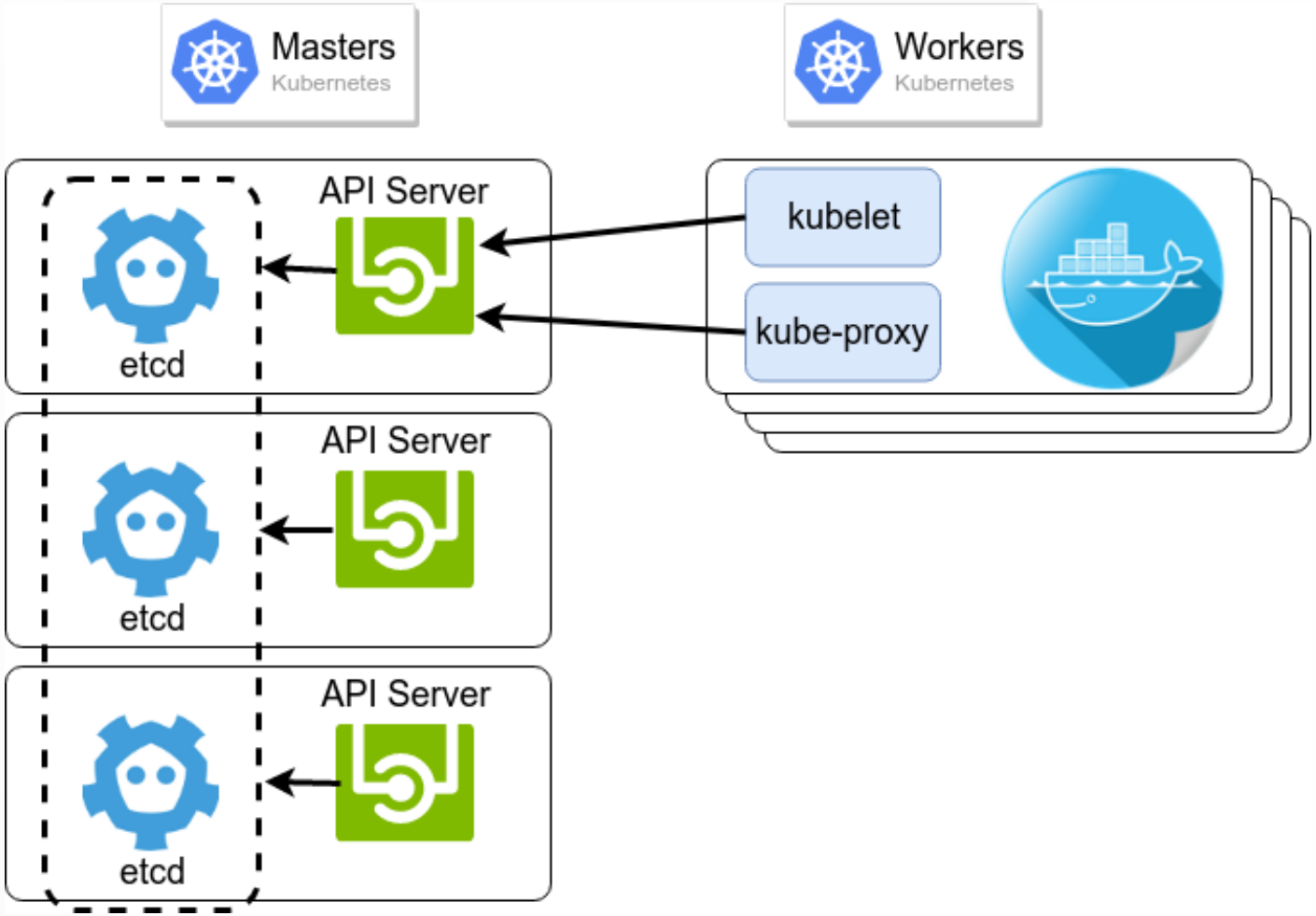
- **à mettre en place dès le début** du cycle de développement
- plus difficile à appliquer *a posteriori* (sauf à tout bloquer)

Pour auditer le RBAC :

- `kubectl auth can-i`
- `kubectl who-can`
- et plein d'autres

“C’est toujours la faute du réseau”

Architecture simplifiée de Kubernetes



Du TLS partout

Tous les flux doivent être chiffrés, *en particulier ceux de Kubernetes* lui-même (api-server, etcd, ...)

Point Captain Obvious : Si les flux ont été chiffrés, il sera plus difficile de récupérer des identifiants

Pas d'APIs sur Internet !

> 2000 Docker engines are insecurely exposed to the Internet unit42 : Docker API + Graboid

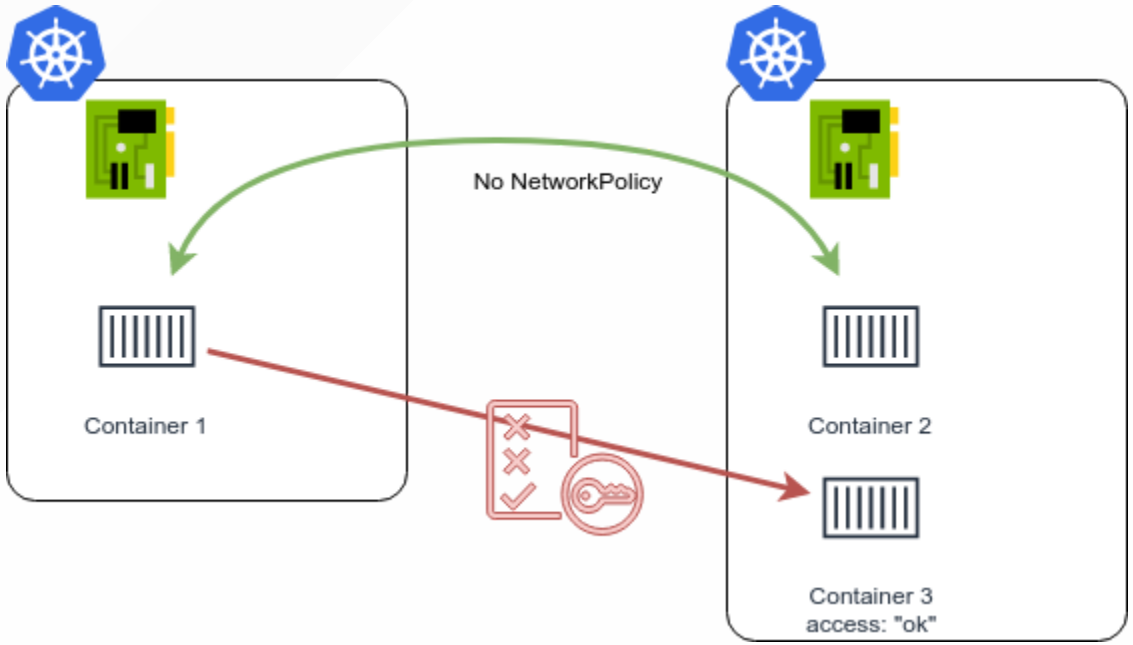
> [...] but it was possible to connect from....the Internet 4armed : etcd + Digital Ocean

> our coworker’s server was also publicly exposing the kubelet ports Handy + kubelet

Ajouter des Network Policies

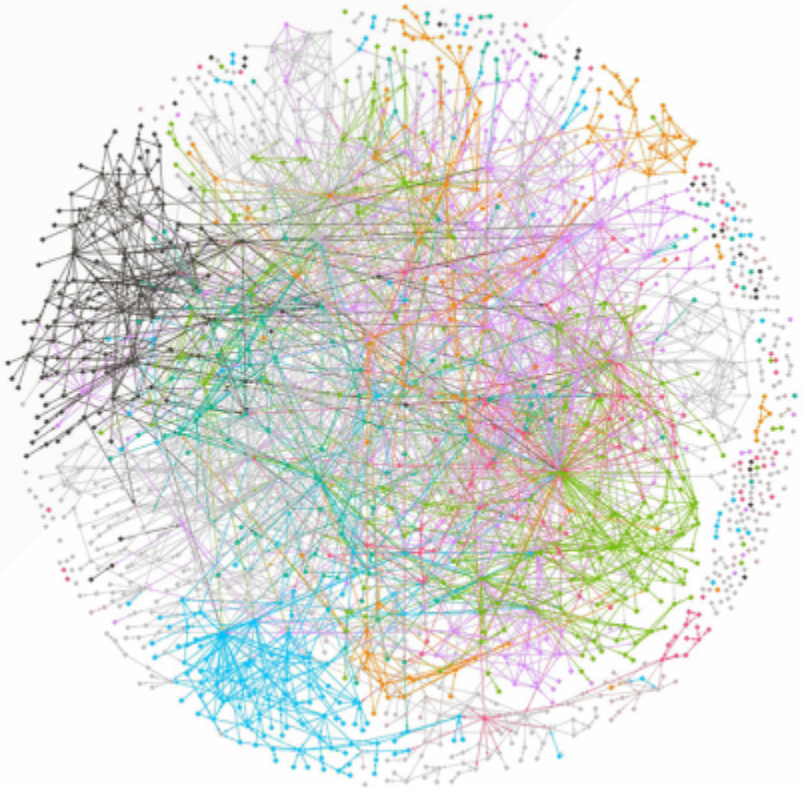
Par défaut, Kubernetes *autorise tout container à se connecter à n'importe quel autre* **#OpenBar**

```yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: access-nginx
spec:
  podSelector:
    matchLabels:
      app: nginx
  ingress:
  - from:
    - podSelector:
        matchLabels:
          access: "true"
```



Sac de nouilles avec les Network Policies

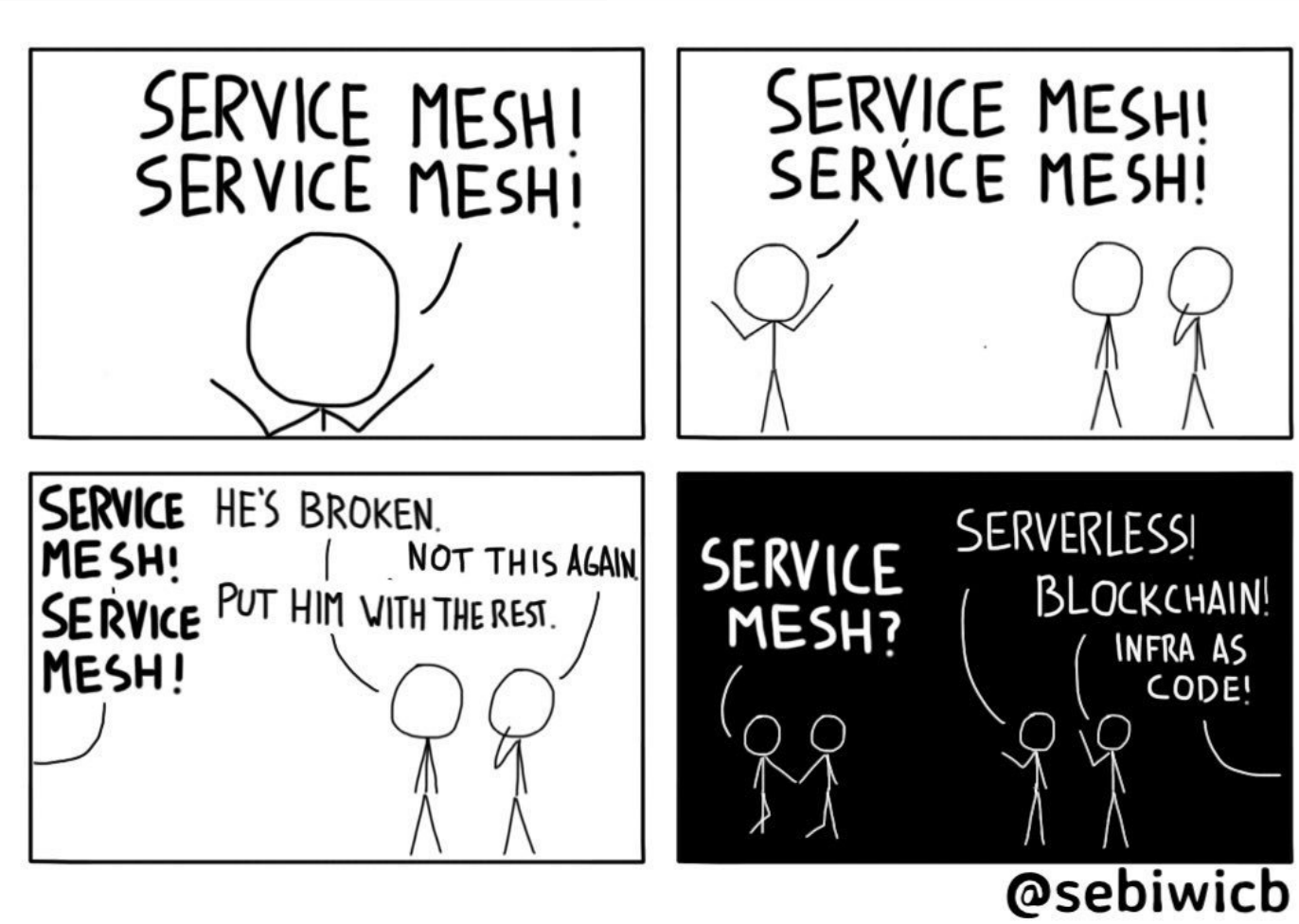
Monzo Bank a mis en place des Network Policies pour la totalité de ses 1500 microservices :



Service Mesh !

Mettre en place des **Network Policies** peut être complexe...

... mais on peut faire encore plus complexe !



Service Mesh

Déléguer beaucoup d'aspects réseau+sécu au Service Mesh :

- gestion TLS
- firewalling / ACL
- analyse temps réel des attaques
  - audit/forensics, DDOS mitigation, ...

Sécuriser les containers

It's secure

Sysadmins/Devs: "It's secure because it's in a container"

Hackers:

Pas de container exécuté en tant que Root !

Kubernetes utilise (pour l'instant) la table des users ID de l'hôte

binaire lancé en tant que **root** = binaire lancé en root sur l'hôte k8s

[Kubecon EU 2018: The route to Rootless Containers]

user@pc:~$ sudo whoami

JW Player

Un des services de supervision (Weave Scope) avait été créé avec des privilèges et accessible depuis le net

> Our deployment was missing the annotation to make the load balancer internal
> The `weave-scope` container is running with the `--privileged` flag
> Files on the root file system were mounted onto the container
> Containers are run as the `root` user.

[How A Cryptominer Made Its Way in our k8s Clusters]

Pod Security Policy

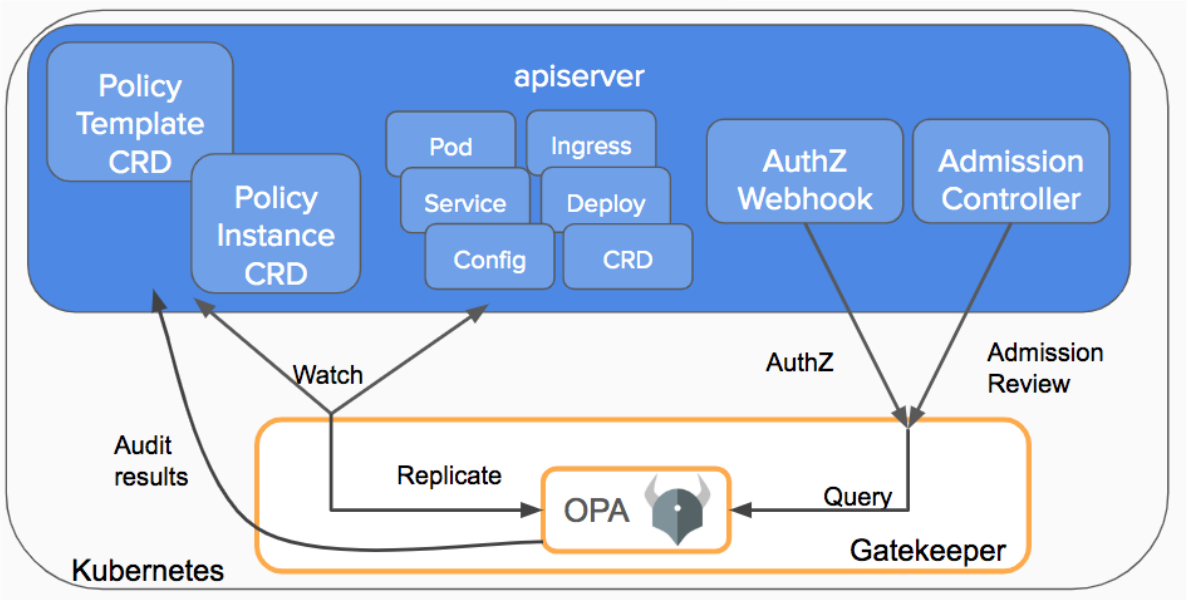
Kubernetes permet l'ajout de politiques de conformités, notamment dans le but d'imposer des règles pour les Pods

```
# Required to prevent escalations to root.
allowPrivilegeEscalation: false
runAsUser:
  # Require the container to run without root privileges.
  rule: 'MustRunAsNonRoot'
```

- Meetup Enix Jpetazzo - escalation via hostPath Volume

~~Pod Security Policy~~ DEPRECATED

Dépréciées depuis Kubernetes 1.21
A remplacer par OPA (Open Policy Agent)



[Vos politiques de conformité sur Kubernetes avec OPA et Gatekeeper]

static scan

Des CVE sortent sur **NodeJS**, **.Net** et autre **JVM** toutes les semaines

Les images de bases de vos containers sont bourrées de failles

clair

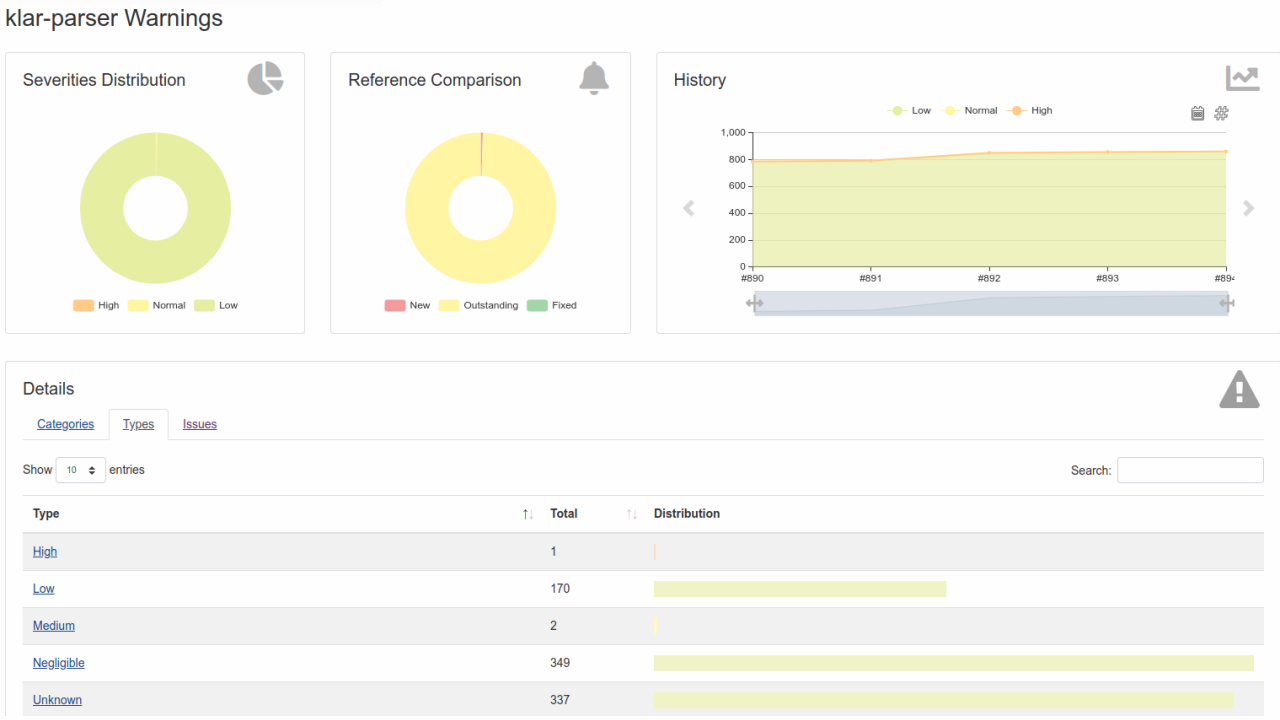
anchore

trivy ...

Concrètement

Interface affichant les failles détectées sur chaque image

Rajouter des **quality gates** côté *Intégration Continue* pour bloquer les images qui ne répondent pas aux exigences de sécurité



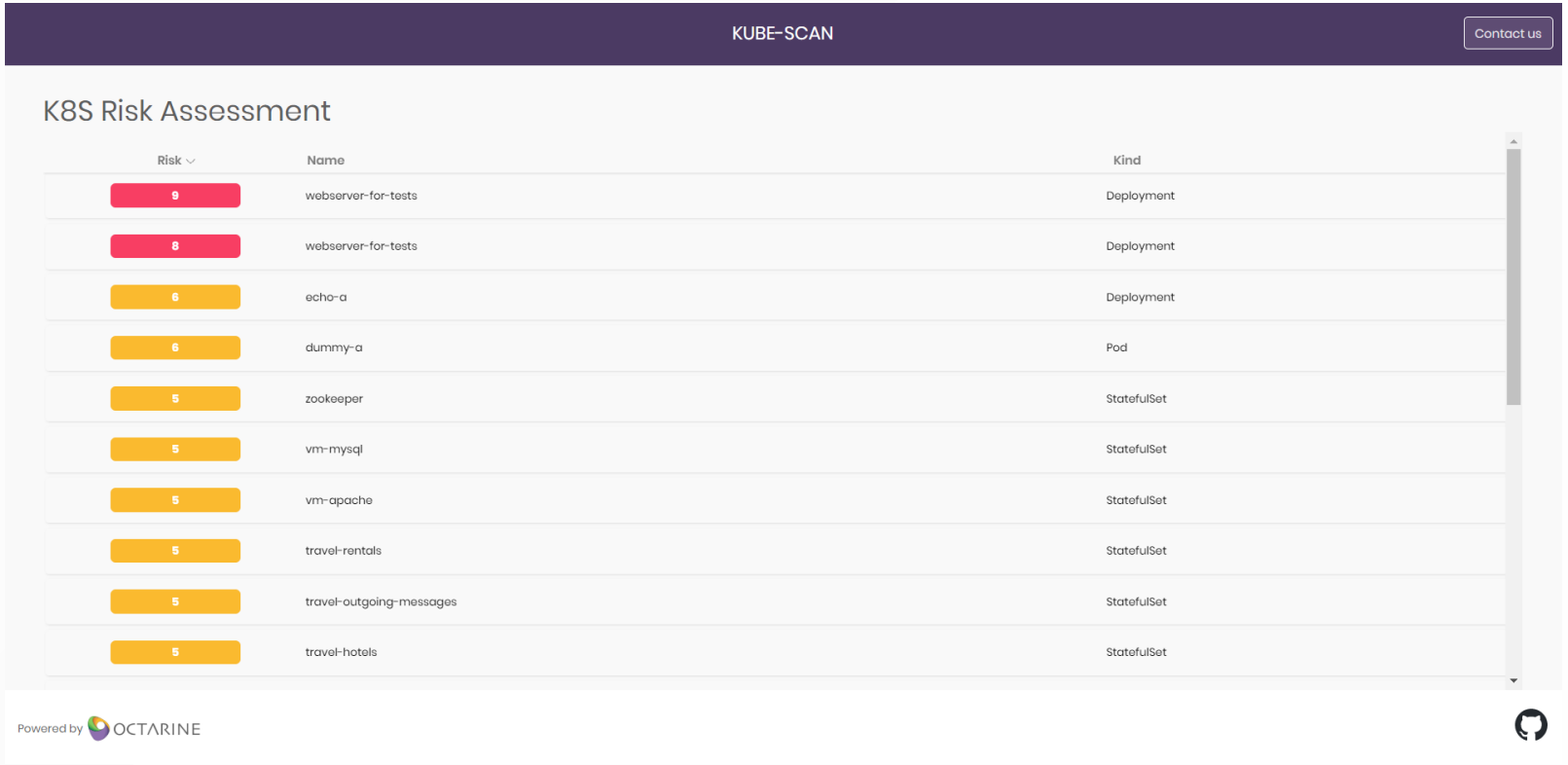
Réduire la surface d'attaque

Limiter l'impact d'une compromission :

- Le moins de dépendances possibles
- Ne pas ajouter des binaires utiles aux attaquants
  - oubliez `ping`, `traceroute`, `gcc`, ...
- Multistage build (images build vs images run)
- **Pas de shell !**

Vérifiez que vos applications

- kubeaudit (audit des app déployées)
- kube-scan (risk assessment du type CVSS)



runtime scan

Il existe aussi des *Intrusion Detection System* pour Kubernetes

> Falco is an open source project for intrusion and abnormality detection for Cloud Native platforms

> Cilium: eBPF-based Networking, Observability, and Security

“Pourquoi *GCC* tourne dans mon container ?”

Utilise des programmes BPF côté kernel pour détecter des comportements anormaux

```
sysdig:~ $ sudo falco
Sat Jan 13 07:11:15 2018: Falco initialized with configuration file /etc/falco/falco.yaml
Sat Jan 13 07:11:15 2018: Parsed rules from file /etc/falco/falco_rules.yaml
Sat Jan 13 07:11:15 2018: Parsed rules from file /etc/falco/falco_rules.local.yaml
07:12:19.322256631: Notice A shell was spawned in a container with an attached terminal (user=root ex
ample1_wordpress_1 (id=25ee73aaf85c) shell=bash parent=<NA> cmdline=bash  terminal=34816)
07:12:24.571565986: Warning Sensitive file opened for reading by non-trusted program (user=root name=
cp command=cp /etc/shadow /var/www/html file=/etc/shadow parent=bash gparent=<NA> ggparent=<NA> gggpa
rent=<NA>)
07:12:36.141717272: Error File below known binary directory renamed/removed (user=root command=mv /bi
n/ls /bin/ls.old operation=rename file=<NA> res=0 oldpath=/bin/ls newpath=/bin/ls.old )
```

[Cilium Uncovering a Sophisticated Kubernetes Attack in Real-Time]

Et l'infra dans tout ça ?

Kubernetes Security Audit

Comme tout logiciel, Kubernetes a des failles !

Aout 2019 : la CNCF a commandé un audit du code de Kubernetes

- Commencé sur un périmètre restreint
- Généralisé à tous les nouveaux composants entrant dans la CNCF
- A permis de déceler 37 vulnérabilités

D'autres failles dans Kubernetes

Trois autres grosses CVE sont sorties "récemment"

- **2018** / faille dans l'API server
  - ZDnet : La première grosse faille de sécurité est là (API server)
- **2019** / faille dans RunC pour sortir du container
  - Faille dans RunC (Docker, Kubernetes et Mesos concernés)
- **2020** / faille dans kubernetes controller manager
  - When it’s not only about a Kubernetes CVE…

Moralité : mettez à jour régulièrement !

... en vrai, c'est pas forcément simple

[Kubecon EU 2018 - Zalando Continuously Deliver your K8s Infra]

Vérifiez AUSSI que votre cluster respecte les bonnes pratiques

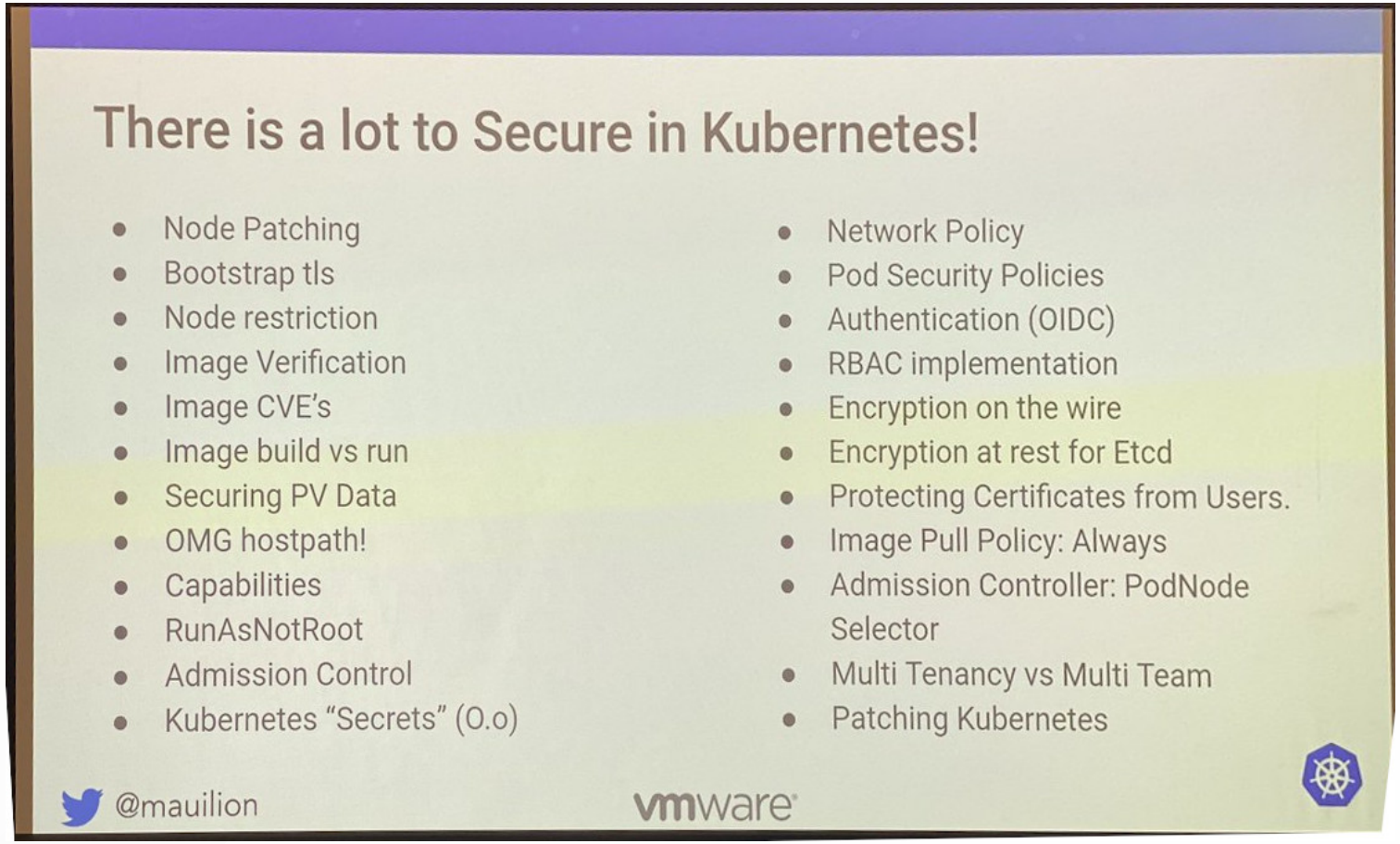
- kube-hunter (scan de vulnérabilité triviales)
- kube-bench (benchmark CSI de votre installation)

```
[INFO] 1 Master Node Security Configuration
[INFO] 1.1 API Server
[FAIL] 1.1.1 Ensure that the --allow-privileged argument is set to false (Scored)
[FAIL] 1.1.2 Ensure that the --anonymous-auth argument is set to false (Scored)
[PASS] 1.1.3 Ensure that the --basic-auth-file argument is not set (Scored)
[PASS] 1.1.4 Ensure that the --insecure-allow-any-token argument is not set (Scored)
[FAIL] 1.1.5 Ensure that the --kubelet-https argument is set to true (Scored)
[PASS] 1.1.6 Ensure that the --insecure-bind-address argument is not set (Scored)
[PASS] 1.1.7 Ensure that the --insecure-port argument is set to 0 (Scored)
[PASS] 1.1.8 Ensure that the --secure-port argument is not set to 0 (Scored)
[FAIL] 1.1.9 Ensure that the --profiling argument is set to false (Scored)
[FAIL] 1.1.10 Ensure that the --repair-malformed-updates argument is set to false (Scored)
[PASS] 1.1.11 Ensure that the admission control policy is not set to AlwaysAdmit (Scored)
[FAIL] 1.1.12 Ensure that the admission control policy is set to AlwaysPullImages (Scored)
[FAIL] 1.1.13 Ensure that the admission control policy is set to DenyEscalatingExec (Scored)
[FAIL] 1.1.14 Ensure that the admission control policy is set to SecurityContextDeny (Scored)
[PASS] 1.1.15 Ensure that the admission control policy is set to NamespaceLifecycle (Scored)
[FAIL] 1.1.16 Ensure that the --audit-log-path argument is set as appropriate (Scored)
[FAIL] 1.1.17 Ensure that the --audit-log-maxage argument is set to 30 or as appropriate (Scored)
[FAIL] 1.1.18 Ensure that the --audit-log-maxbackup argument is set to 10 or as appropriate (Scored)
[FAIL] 1.1.19 Ensure that the --audit-log-maxsize argument is set to 100 or as appropriate (Scored)
[PASS] 1.1.20 Ensure that the --authorization-mode argument is not set to AlwaysAllow (Scored)
[PASS] 1.1.21 Ensure that the --token-auth-file parameter is not set (Scored)
[FAIL] 1.1.22 Ensure that the --kubelet-certificate-authority argument is set as appropriate (Scored)
```

Exemple d'attaque via Unauthenticated Kubelet

“Promis, demain, je sécurise”

There is a lot to Secure



[Source: Kubernetes Security / Duffie Cooley]

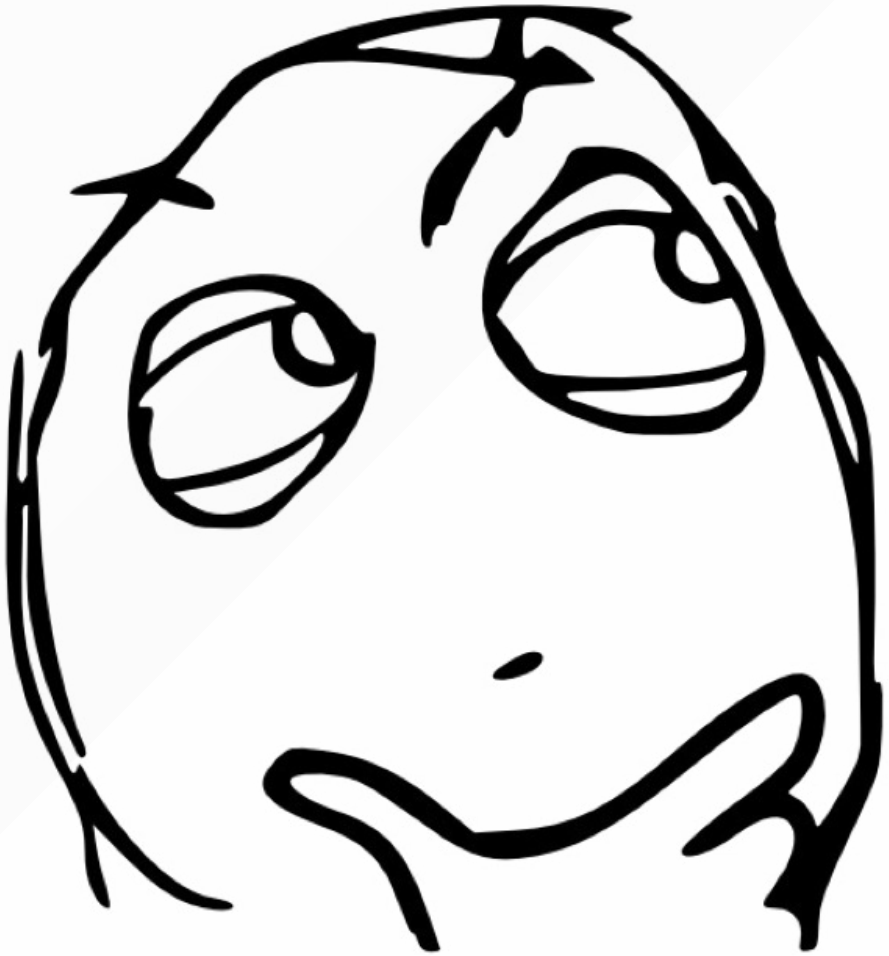
Conclusion

- Ne déployez pas Kubernetes si vous n'en avez pas besoin !
  - Mais si vous pouvez le faire, faites le !
  - blog : combien de problèmes ces stacks ont générés ?

- Il y a beaucoup de choses à sécuriser dans Kube
  - Formez vos développeurs, pas seulement les Ops !
  - Sécurisez dès le début

That's all folks

Des questions ?



Backup slides

Kubernetes threat matrix

Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Impact
Using Cloud credentials	Exec into container	Backdoor container	Privileged container	Clear container logs	List K8S secrets	Access the K8S API server	Access cloud resources	Data Destruction
Compromised images in registry	bash/cmd inside container	Writable hostPath mount	Cluster-admin binding	Delete K8S events	Mount service principal	Access Kubelet API	Container service account	Resource Hijacking
Kubeconfig file	New container	Kubernetes CronJob	hostPath mount	Pod / container name similarity	Access container service account	Network mapping	Cluster internal networking	Denial of service
Application vulnerability	Application exploit (RCE)		Access cloud resources	Connect from Proxy server	Applications credentials in configuration files	Access Kubernetes dashboard	Applications credentials in configuration files	
Exposed Dashboard	SSH server running inside container					Instance Metadata API	Writable volume mounts on the host	
							Access Kubernetes dashboard	
							Access tiller endpoint	

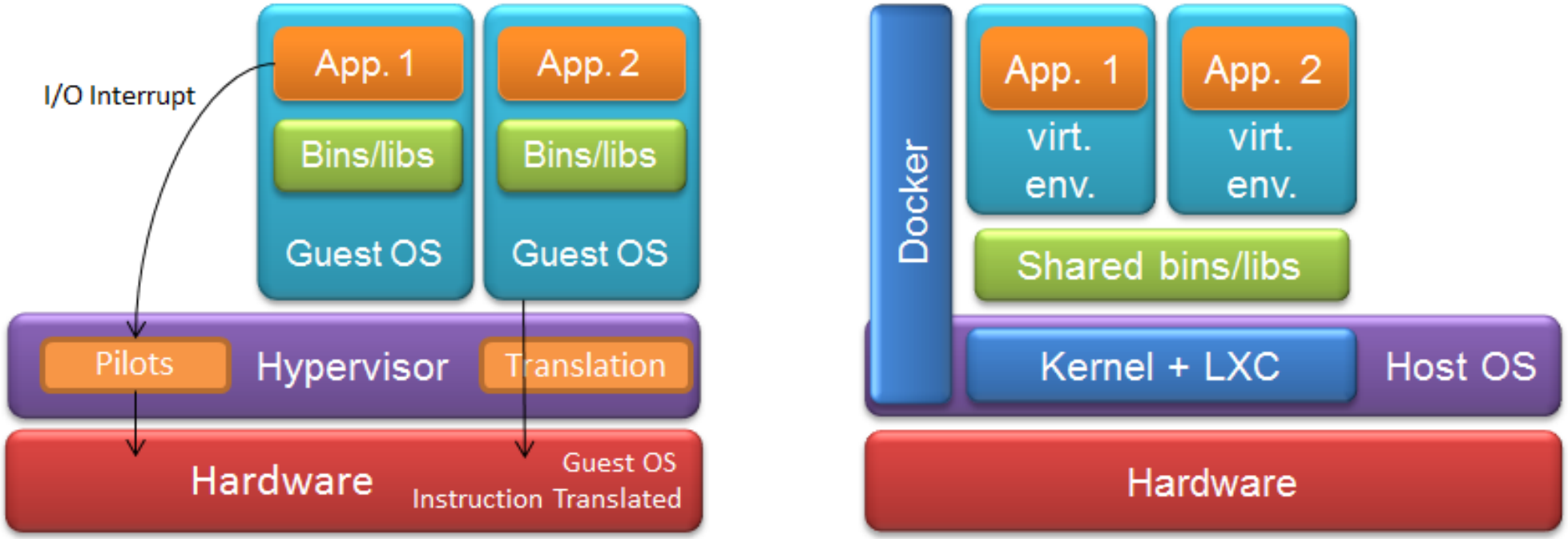
[Microsoft Security Blog]

La mode des containers

> **Outil** qui permet d'empaqueter une application et ses dépendances. Elle pourra être exécuté sur n'importe quel serveur

- Il existe de nombreuses implémentations des containers
-  est très utilisé depuis quelques années
  - utilise des fonctionnalités du kernel Linux
  - fourni une interface "simple" et un magasin d'images

Le container, ce n'est pas une VM !



Pourquoi utiliser les containers Docker ?

Pratique si on déploie souvent de "petites" applications, en cycle (très) courts

L'application devient immuable

- Si on veut l'upgrader ou changer sa configuration :
  - on ne modifie pas le container (source d'erreur)
  - on déploie la nouvelle version et on supprime l'ancienne

[blog.zwindler.fr / Should we have containers ?]

Sources

Les best practices

- Kubernetes.io : 11 ways not to get hacked
- CNCF : Kubernetes security best practices
- Rancher : More Kubernetes best practices
- Stackrox
- Jerry Jalava : Kubernetes Security Journey
- Workshop Sécuriser son Kubernetes au DevFest Nantes 2019
- Duffie Cooley (VMware) : Kubernetes Security

Les outils pour durcir Kube (1/2)

- Scan de vulnérabilité triviales : kube-hunter
- Benchmark CSI de votre installation : kube-bench
- OnePolicyAgent
- Audit des app déployées : kubeaudit
- "Risk assessment" du type CVSS : kube-scan

Les outils pour durcir Kube (2/2)

- Analyse statique : Clair
- Analyse statique : Anchore
- Analyse statique : Trivy
- Analyse runtime : Falco
- CNI + analyse runtime : Cilium
- SELinux, Seccomp, Falco, a technical discussion

Les failles de sécu récentes de K8s (&+)

- Faille dans RunC (Docker, Kubernetes et Mesos concernés)
- Liste des CVE Kubernetes
- ZDnet : La première grosse faille de sécurité est là (API server)
- L'exploit pour la faille dans l'API Server

Les sociétés hackées dans la presse

- 2018 : Cryptojacking chez Tesla
- 2019 : Cryptojacking chez jwplayer
- Plus d'infos sur le cryptominer XMrig
- ZDNET : des hackers utilisent les API de management de Docker exposées sur le net
- zdnet - Microsoft découvre un gang de cryptomining détournant des clusters Kubernetes
- 4armed : Server-Side Request Forgery + **etcd** accessible depuis Internet chez Digital Ocean
- Un cluster perso d'un employé de Handy laisse son kubelet ouvert

Kubernetes Security Audit

- Audit du code en aout 2019 : article principal
- Audit du code en aout 2019 : audit en lui même

Autre

- Outil d'audit des rôles
- Une liste d'outils permettant d'auditer le RBAC dans Kubernetes
- Une image XMRig (Monero) sur le Dockerhub
- Twitter : "There is a lot to Secure in Kubernetes"
- DNS Spoofing on Kubernetes Clusters
- Docker and Kubernetes Reverse shells
- Exploiter un Tomcat Manager non sécurisé
- Backdoor dans webmin